LAB Logbook

Lab 1

Lab Logbook Requirement:

1) Create a vector using np.arange.

Determine the number of the vector elements using the following method: Take the last two digits from your SID. It should be from 00 to 99. If this number is 10 or more, it becomes the required number of the vector elements. If it is less than 10, add 100 to your number.

For example, if your SID is 2287467, and the last two digits are 67, which is greater than 10. The required number is 67. If your SID is 2287407, and the last two digits are 07, which is less than 10. The required number is 107.

Then,

2. Change matrix a to 2-d array with 1 row. Print the array. You should have the two sets of brackets for a 2-d array with one row.
3. Save it in another array. Print the array.
4. Check the shape attribute value.
5. Add the code and result to your Lab Logbook

NOTE: DON'T FORGET TO SAVE AND BACK UP YOUR COMPLETED JUPYTER NOTEBOOK AND LAB LOGBOOK ON GITHUB OR ONEDRIVE.

```
In [69]:  SID = 2285050
```

```
In [72]:  number_of_vector_elements = 50
          vector = np.arange(number_of_vector_elements)
          print("Vector: ", vector)

Vector:  [ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
 48 49]
```

```
In [73]:  _2d_array = vector.reshape(1, -1)
          print("2D array with one row: ", _2d_array)

2D array with one row:  [[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
 48 49]]
```

```
In [74]:  another_array = _2d_array.copy()
          print("Another array: ", another_array)

Another array:  [[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
 48 49]]
```
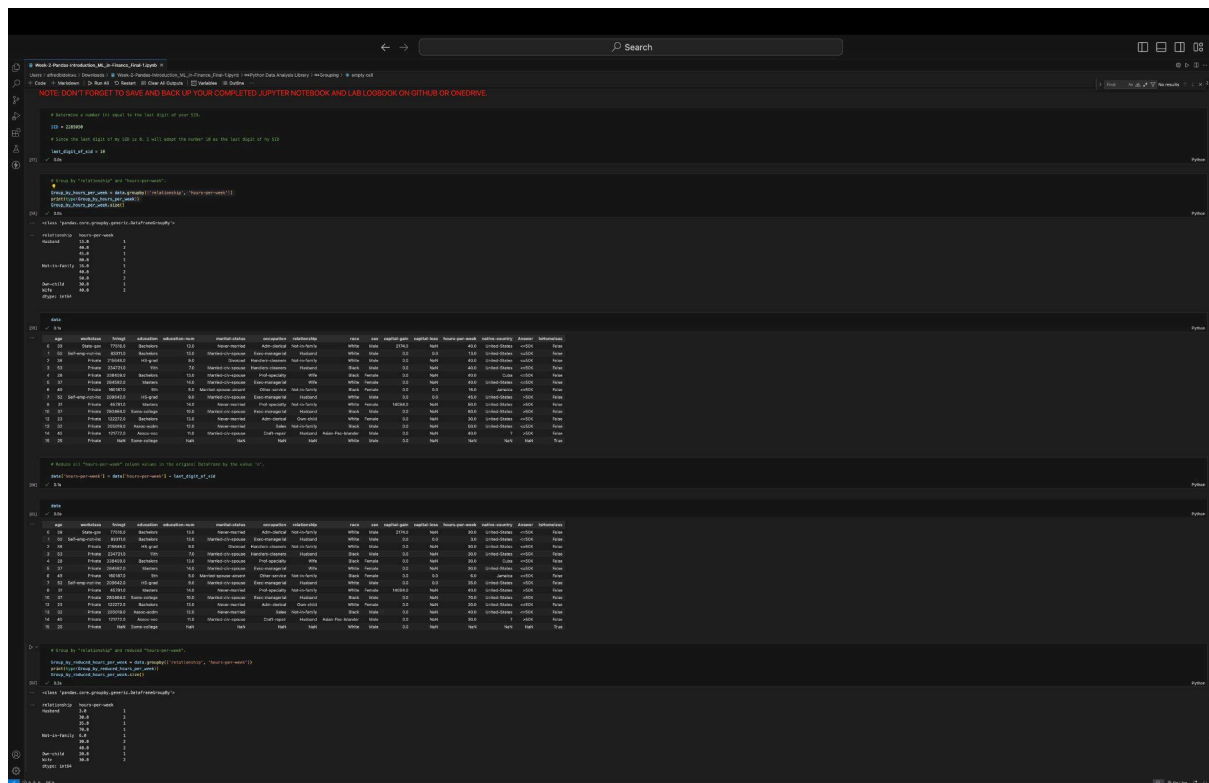
```
In [75]:  shape_value = another_array.shape
          print("Shape: ", shape_value)

Shape:  (1, 50)
```
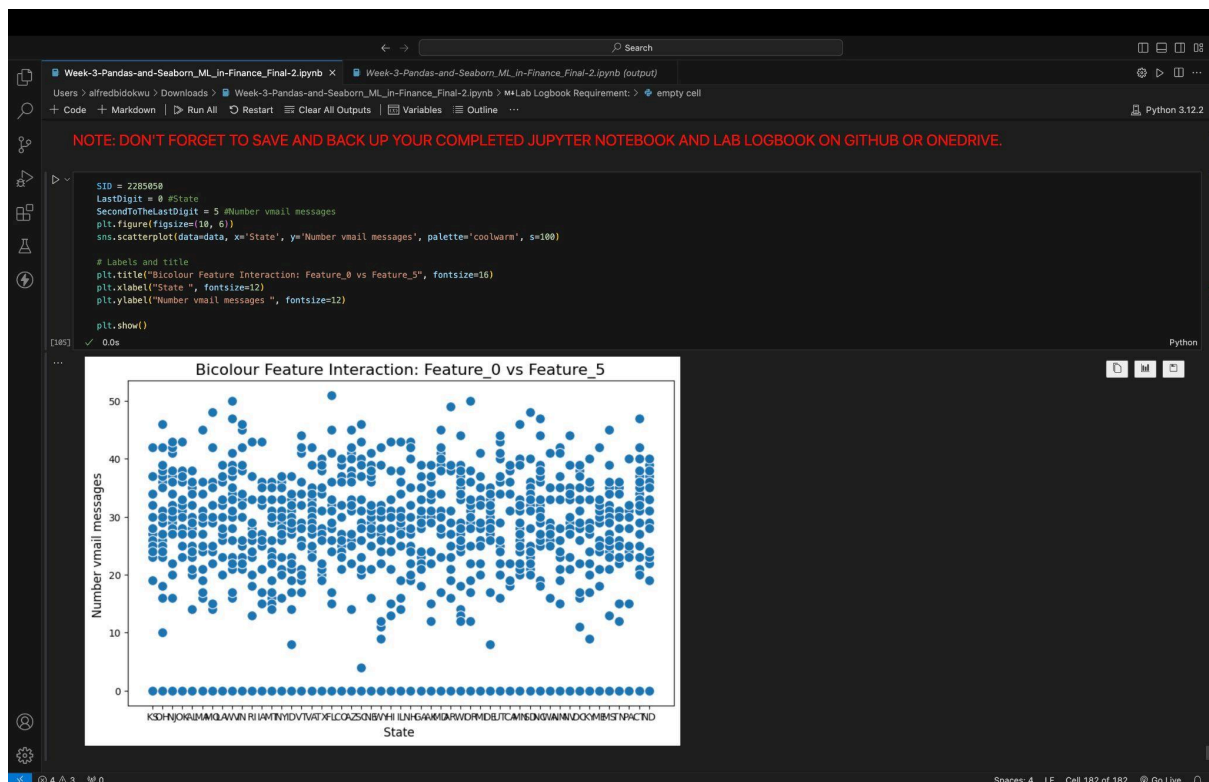
```
In [ ]:
```

Lab 2

Lab 3

Lab 4

Lab 5

Lab 6

Lab Logbook Requirement:

1. Plot the price chart of the part of the whole dataset 'High_Bid' and 'Low_Bid' prices using plotly library.
2. The start point should equal the 5 last digits of your SID Number.
3. The time period (in minutes) should equal the 3 last digits of your SID Number.
4. Please only add a print screen of your code and final graph to your Lab Logbook.

NOTE: DON'T FORGET TO SAVE AND BACK UP YOUR COMPLETED JUPYTER NOTEBOOK AND LAB LOGBOOK ON GITHUB OR ONEDRIVE.

My SID is 2285050

Therefore my last 5 digits are 85050 while my last 3 digits are 050

```python
data2.iloc[85050:85100][['High_Bid', 'Low_Bid']].iplot(
                        mode='lines+markers',
                        xTitle='Prices', yTitle='Numbers',
                        title='GOLD ')
```


Lab 7

```python
print(model.summary())
```
[96] ✓ 0.0s

Model: "sequential"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 100)	47,600
dense (Dense)	(None, 2)	202

Total params: 143,408 (560.19 KB)

Trainable params: 47,802 (186.73 KB)

Non-trainable params: 0 (0.00 B)

Optimizer params: 95,606 (373.46 KB)

None

```
print("Mean squared error (mse): %.9f " % (scores[0]))

print("Mean absolute error (mae): %.9f " % (scores[1]))
```
[92] ✓ 0.0s

... Mean squared error (mse): 0.000025137
     Mean absolute error (mae): 0.003919592
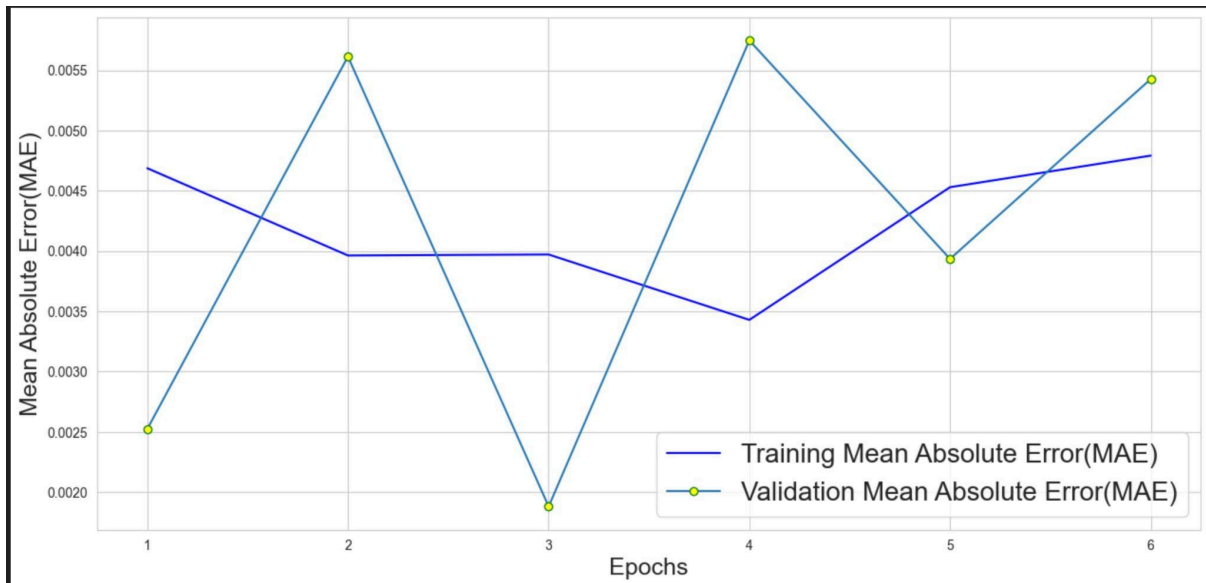
∨ Comparism between the MSE & MAE in the practical session and the MSE & MAE of the Assignment

Mean Squared Error (MSE): Practical session: 0.000052403 Assignment: 0.000025137 The MSE is significantly lower in the assignment model. MSE measures the average squared differences between predicted and actual values, which gives more weight to larger errors. The lower MSE in the assignment implies that the predictions of the assignment model are closer to the actual values, with fewer large errors compared to the practical model.

Mean Absolute Error (MAE): Practical session: 0.005791005 Assignment: 0.003919592 The MAE is also lower in the assignment model. MAE measures the average magnitude of errors in the predictions, without considering their direction. A lower MAE indicates that, on average, the assignment model's predictions deviate less from the actual values than the practical model's predictions.

Implications: Improved Model Performance: The lower MSE and MAE in the assignment model suggest that the changes in model parameters have led to improved accuracy and predictive capability. Fewer Large Errors: Since MSE is more sensitive to large errors, the lower value for the assignment model indicates that it is better at avoiding significant deviations from actual values. Refinement of Parameters: The parameter adjustments made in the assignment have likely enhanced the model's ability to capture the underlying patterns in the data more effectively.

Therefore, the assignment model is quantitatively better than the practical model, both in terms of average error magnitude (MAE) and error sensitivity (MSE).

Lab 8

CNN METRICS:

```
print("Mean squared error (mse): %.9f " % (scores[0]))
```

Mean squared error (mse): 0.009019411

```
print("Mean absolute error (mae): %.9f " % (scores[1]))
```

Mean absolute error (mae): 0.073248647

LSTM METRICS:

```
print("Mean squared error (mse): %.9f " % (scores[0]))
```

Mean squared error (mse): 7.449756145

```
print("Mean absolute error (mae): %.9f " % (scores[1]))
```

Mean absolute error (mae): 1.905555487

MLP METRICS:

```
    print("Mean squared error (mse): %.9f " % (scores[0]))
✓   0.0s
```

Mean squared error (mse): 0.005189301

```
    print("Mean absolute error (mae): %.9f " % (scores[1]))
✓   0.0s
```

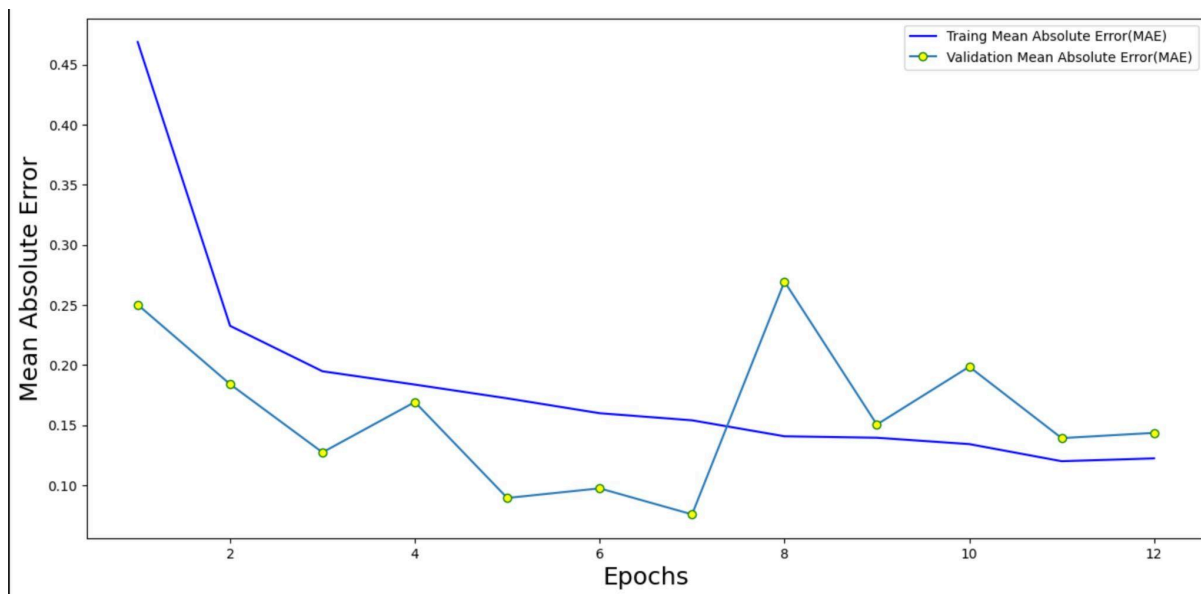Mean absolute error (mae): 0.051906992

Lab 9

TEST WEEK

Lab 10

```
print("Mean squared error (mse): %.9f " % (scores[0]))
```
✓ 0.0s

Mean squared error (mse): 0.005189301

```
print("Mean absolute error (mae): %.9f " % (scores[1]))
```
✓ 0.0s

Mean absolute error (mae): 0.051906992

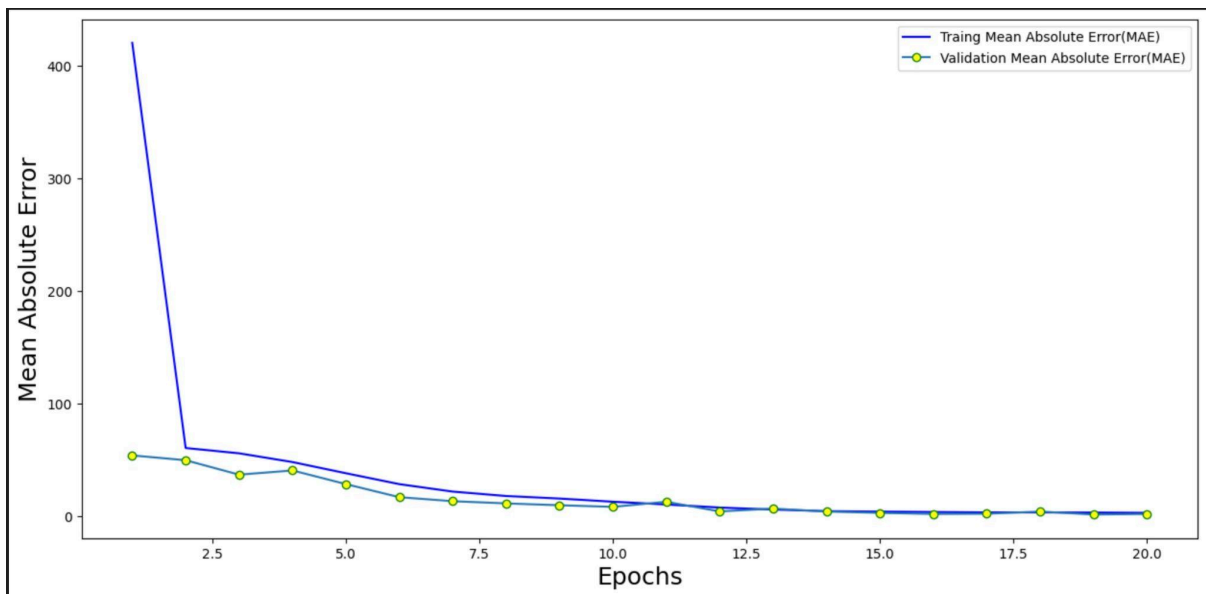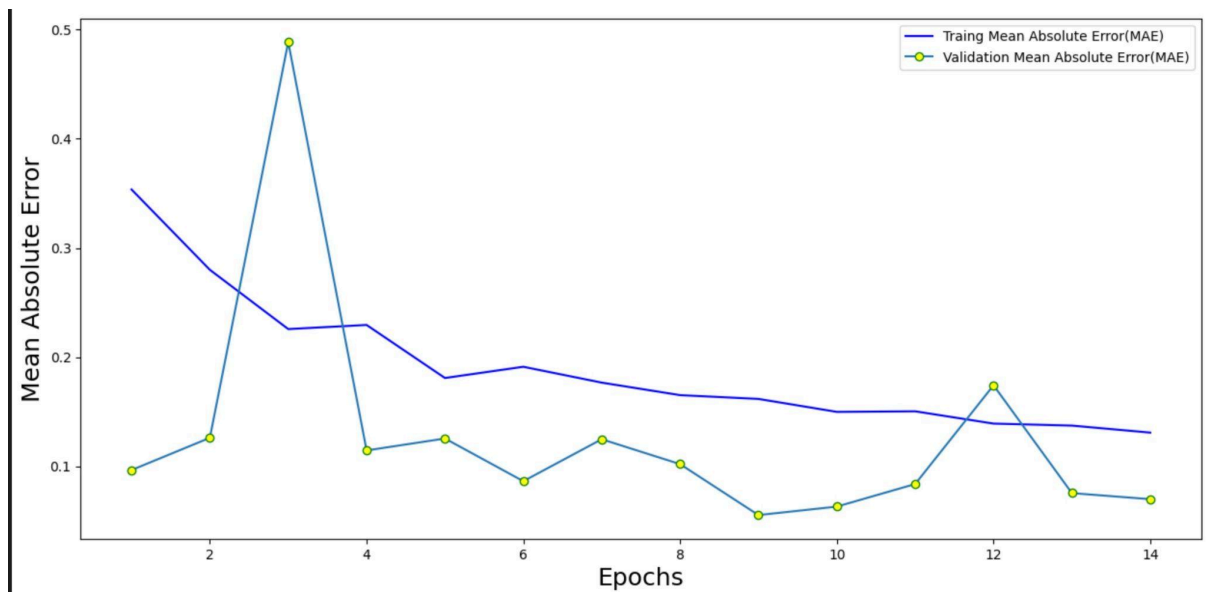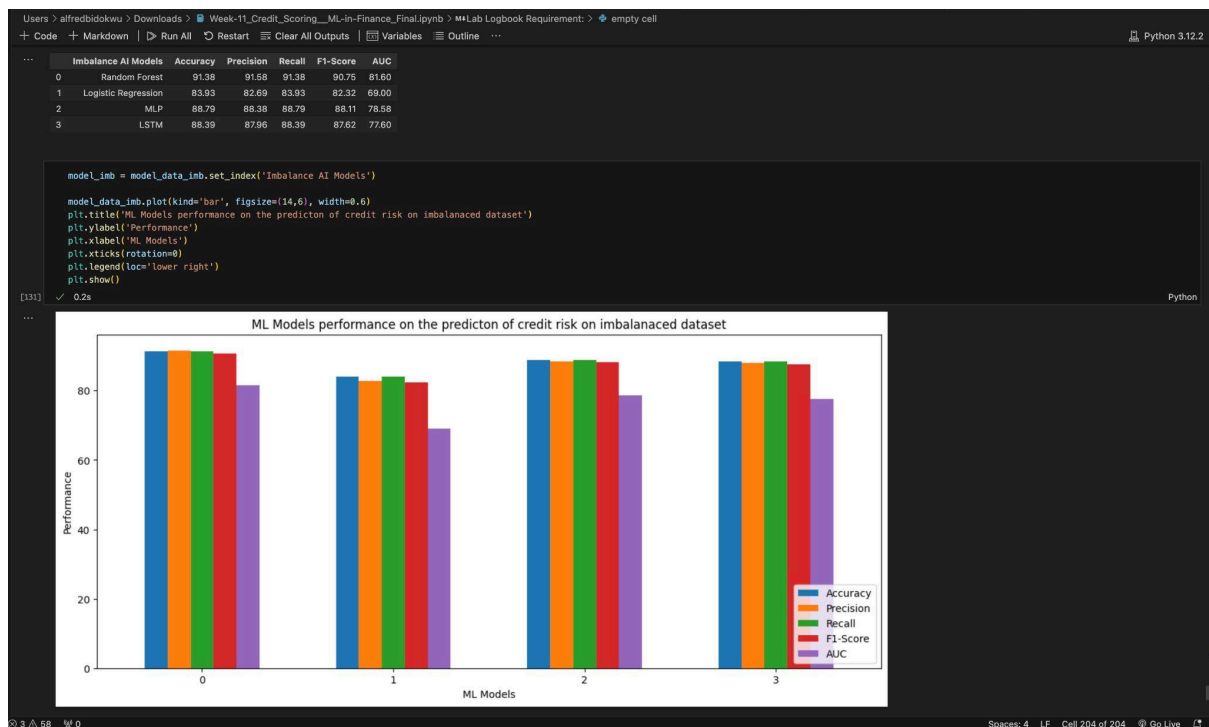# Lab 11

	Imbalance AI Models	Accuracy	Precision	Recall	F1-Score	AUC
0	Random Forest	91.38	91.58	91.38	90.75	81.60
1	Logistic Regression	83.93	82.69	83.93	82.32	69.00
2	MLP	88.79	88.38	88.79	88.11	78.58
3	LSTM	88.39	87.96	88.39	87.62	77.60

```python
model_imb = model_data_imb.set_index('Imbalance AI Models')

model_data_imb.plot(kind='bar', figsize=(14,6), width=0.6)
plt.title('ML Models performance on the predicton of credit risk on imbalanaced dataset')
plt.ylabel('Performance')
plt.xlabel('ML Models')
plt.xticks(rotation=0)
plt.legend(loc='lower right')
plt.show()
```
[131] ✓ 0.2s

ML Models performance on the predicton of credit risk on imbalanaced dataset

Lab 12